

8.2 课堂笔记 29

本次作业

本次作业完成情况作为此次选拔成绩的一部分，务必认真完成

作业一：修正 T1、T2、T3 到满分，T4 选做。

T1: P5686

T2: P1018

T3: P1073

T4: P1967

作业二：完成以下题单，8.7 晚统计洛谷做题记录，并检查代码是否自己完成。

绿签以下：

P1017

P1024

P1031

P1089

P2196

绿签：

P1021

P1016

P1541

P1525

P1073

P1072

初赛练习

CSP-S 第十三套初赛模拟题解析

4、

类别	排序方法	时间复杂度			空间复杂度	稳定性
		平均情况	最好情况	最坏情况	辅助存储	
插入排序	直接插入	$O(n^2)$	$O(n)$	$O(n^2)$	$O(1)$	稳定
	Shell排序	$O(n^{1.3})$	$O(n)$	$O(n^2)$	$O(1)$	不稳定
选择排序	直接选择	$O(n^2)$	$O(n^2)$	$O(n^2)$	$O(1)$	不稳定
	堆排序	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(1)$	不稳定
交换排序	冒泡排序	$O(n^2)$	$O(n)$	$O(n^2)$	$O(1)$	稳定
	快速排序	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(n^2)$	$O(n\log_2 n)$	不稳定
归并排序		$O(n\log_2 n)$	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(n)$	稳定
基数排序		$O(d(r+n))$	$O(d(n+rd))$	$O(d(r+n))$	$O(rd+n)$	稳定

注：基数排序的复杂度中， r 代表关键字的基数， d 代表长度， n 代表关键字的个数。

排序优势：

- 1、值域很小时，桶排序最优
- 2、数据量小时，选择插入排序和选择排序
- 3、数据量很大时，要求稳定排序就选归并排序；
不要求稳定还可以选快速排序和堆排序
- 4、空间复杂度：堆排序 < 快速排序 < 归并排序
- 5、初态已基本有序，选择插入排序和冒泡排序
- 6、稳定：插入排序、冒泡排序、归并排序、桶排序

5、

操作系统：管理系统资源的分配、程序控制和人机交互
系统资源包括处理机、内存、文件、设备

11、

运算顺序：除法>加法>左移>按位与

口诀：

指针最优，单目运算优于双目运算。如正负号。

先算术运算，后移位运算，最后位运算。

逻辑运算最后结合。

12、

n 个点的无向完全图有 $n(n-1)/2$ 条边

n 个点的图无环至多 $n-1$ 条边

15、

Dijkstra 不能求负边权的最短路

复赛练习

1. P2679

解题思路：

三种状态：

- 1、A 取到了哪里
- 2、B 组成到了那里
- 3、k 取到第几种了

可以定义两个状态数组来解决：

1、 $dp[i][j][k]$ 表示 A[i] 必取，则需要分情况讨论：A[i] 是不是单独成一个子串

2、 $f[i][j][k]$ 表示 A[i] 可取可不取，则需要继承 $f[i-1][j][k]$ 和

$dp[i][j][k]$

代码：

```

1  #include<bits/stdc++.h>
2  using namespace std;
3  const int mod=1e9+7;
4  int dp[3][1010][1010]; //表示一定用了s1的某位置字符
5  int f[3][1010][1010]; //表示不一定用了s1的某位置字符
6  char s1[1010],s2[1010];
7  int n,m,t;

9  int main(){
10     cin >> n >> m >> t;
11     cin >> s1+1 >> s2+1;
12     int now=1,past=0; //i, i-1, i-2
13     f[0][0][0]=1; //可选可不选, 初始化为1
14     for(int i=1;i<=n;i++){
15         f[now][0][0]=1; //可选可不选, 初始化为1
16         for(int j=1;j<=m;j++){
17             for(int k=1;k<=t;k++){
18                 if(s1[i]==s2[j]){
19                     //当前字符必须要用, 但是分为"是否独自成一个子串"两种情况
20                     //f[past][j-1][k-1]表示A[i]独自成一个子串
21                     //dp[past][j-1][k]表示与前面的共用一个子串
22                     dp[now][j][k]=(dp[past][j-1][k]+f[past][j-1][k-1])%mod;
23                 }
24                 else{
25                     //当前字符不能用, dp数组自然为0
26                     dp[now][j][k]=0;
27                 }
28             }
29             //f[now][j][k]有两个来源: 用和不用当前字符
30             f[now][j][k]=(f[past][j][k]+dp[now][j][k])%mod;
31         }
32         swap(now,past);
33     }
34     cout << f[past][m][t];
35     return 0;

```

2. P5687

解题思路:

一、暴力写法: 直接最小生成树模板 60 分

二、优化: 既然同一行或者同一列的的权值一样, 那么要连就把某一行或者某一列的全部连完

- 1、边的权值排序
- 2、先把权值最小的某行、某列连起来, 保证连通
- 3、按照权值从小到大的顺序连边

代码:

```

1  #include<bits/stdc++.h>
2  using namespace std;
3  long long n,m,a[300010],b[300010],p,q,row,line;
4  long long ans,sum;
5
6  int main(){
7      cin >> n >> m;
8      for(int i=1;i<=n;i++) cin >> a[i];
9      for(int i=1;i<=m;i++) cin >> b[i];
10     sort(a+1,a+1+n);
11     sort(b+1,b+1+m);
12     ans += (m-1)*a[1]+(n-1)*b[1]; //最小的两个直接连上去
13     sum=n-1+m-1; //sum表示连了多少条边了
14     row=line=1;
15     //当连接的边数为n*m-1时, 构成最小生成树
16     for(p=2,q=2; sum<(n*m-1); ){
17         if(a[p]<=b[q]){
18             ans += a[p]*(m-row);
19             p++;
20             line++;
21
22             sum += m-row;
23         }
24         else{
25             ans += b[q]*(n-line);
26             q++;
27             row++;
28             sum += n-line;
29         }
30     }
31     cout << ans;
32     return 0;

```